EXHIBIT 022

U.S. Patent No. 7,769,893 (Goossens)
"Integrated circuit and method for establishing transactions"

'9893 Patent Claim	Samsung Exynos 1280 System on Chip[1]
4. A method for exchanging messages in an integrated circuit comprising a plurality of modules, the messages between the plurality of modules being exchanged via a network	Without conceding that the preamble of claim 4 of the '9893 Patent is limiting, Samsung Electronics Co., Ltd.'s (hereinafter, "Samsung") Exynos 1280 system on chip (hereinafter, the "Exynos SoC") is an integrated circuit and performs a method for exchanging messages in an integrated circuit comprising a plurality of modules, the messages between the plurality of modules being exchanged via a network, either literally or under the doctrine of equivalents.

[1] The Exynos SoC is charted as a representative product made used, sold, offered for sale, and/or imported by or on behalf of Samsung. The citations to evidence contained herein are illustrative and should not be understood to be limiting. The right is expressly reserved to rely upon additional or different evidence, or to rely on additional citations to the evidence cited already cited herein.

U.S. Patent No. 7,769,893 (Goossens)
"Integrated circuit and method for establishing transactions"

'9893 Patent Claim	Samsung Exynos 1280 System on Chip[1]

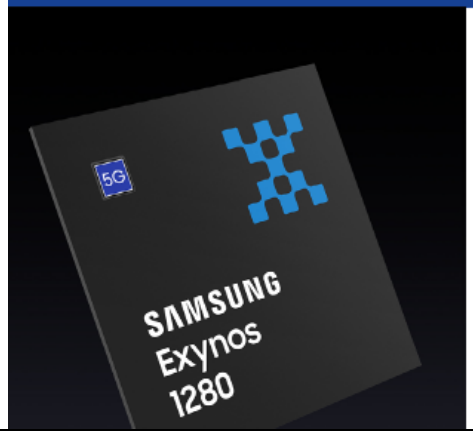
SAMSUNG

Product brief
Create infinite possibilities

Exynos 1280

Highlights

A mobile processor ready for 5G and AI
Advanced ISP and MFC for rich multimedia experience
Powerful octa-core CPU and GPU



5G for all

Exynos 1280 is a mobile processor based on a 64-bit RISC processor. It contains a 5G modem, which is compliant with two types of 5G network (Sub-6GHz and mmWave), as well as all legacy networks. It is built using an advanced 5nm EUV process for high power efficiency.

All-in-one processor for 5G

The Exynos 1280 embedded modem supports both sub-6GHz (Frequency Range

U.S. Patent No. 7,769,893 (Goossens)
"Integrated circuit and method for establishing transactions"

'9893 Patent Claim	Samsung Exynos 1280 System on Chip[1]

https://semiconductor.samsung.com/resources/brochure/Exynos1280.pdf

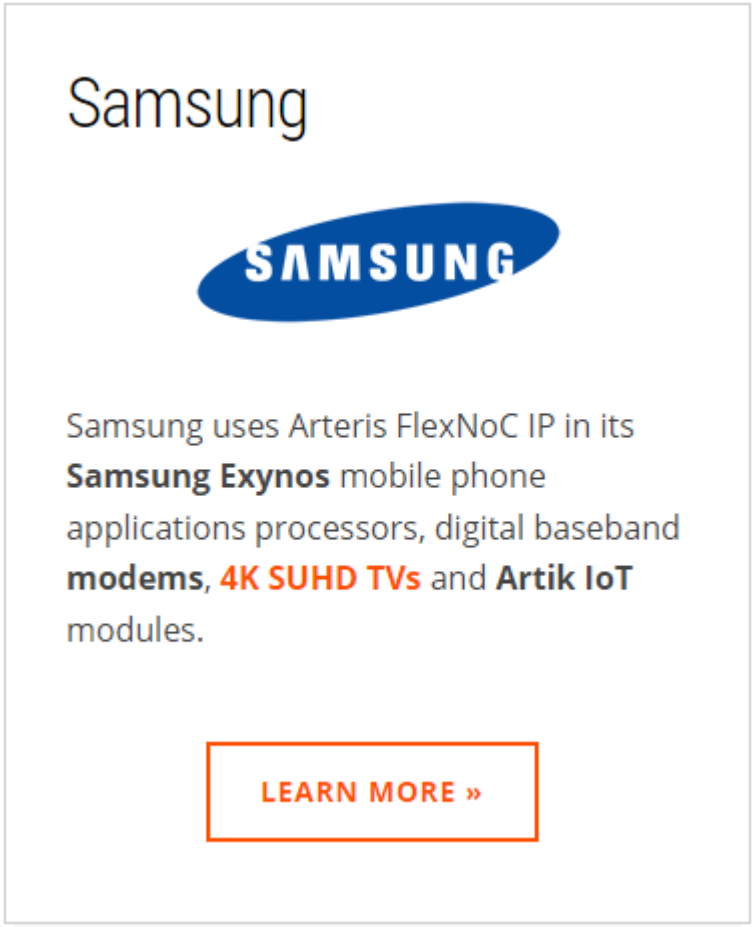
The Exynos SoC comprises a plurality of modules, for example Arm Cortex-A78 core, Cortex-A55 core, Arm Mali-G68 GPU, and AI Engine with NPU:

Specifications

	Exynos 1280
CPU	Cortex®-A78 x 2 + Cortex®-A55 x 6
GPU	Mali™-G68
AI	AI Engine with NPU
Modem	5G NR Sub-6GHz 2.55Gbps (DL) / 1.28Gbps (UL) 5G NR mmWave 1.84Gbps (DL) / 0.92Gbps (UL) LTE Cat.18 6CC 1.2Gbps (DL) / Cat.18 2CC 200Mbps (UL)
Connectivity	WiFi 802.11ac MIMO with Dual-band (2.4/5G), Bluetooth® 5.2, FM Radio Rx
GNSS	Quad-constellation multi-signal for L1 and L5 GNSS
Camera	Up to 108MP in single camera mode, Single-camera 32MP @30fps
Video	4K 30fps encoding and decoding
Display	Full HD+@120Hz
Memory	LPDDR4x
Storage	UFS v2.2
Process	5nm

https://semiconductor.samsung.com/resources/brochure/Exynos1280.pdf

U.S. Patent No. 7,769,893 (Goossens)
"Integrated circuit and method for establishing transactions"

'9893 Patent Claim	Samsung Exynos 1280 System on Chip[1]
	The Exynos SoC utilizes Arteris network on chip interconnect technology, and/or a derivative thereof, (collectively, the "Arteris NoC") for exchanging messages:  https://web.archive.org/web/20210514110614/https://www.arteris.com/customers

U.S. Patent No. 7,769,893 (Goossens)
"Integrated circuit and method for establishing transactions"

'9893 Patent Claim	Samsung Exynos 1280 System on Chip[1]
	Arteris IP FlexNoC® Interconnect Licensed by Samsung's System LSI Business for Digital TV Chips by **Kurt Shuler**, on April 23, 2019 CAMPBELL, Calif. –April 23, 2019– Arteris IP, the world's leading supplier of innovative, silicon-proven network-on-chip (NoC) interconnect semiconductor intellectual property, today announced that Samsung's System LSI Business has renewed multiple Arteris IP FlexNoC Interconnect licenses for use in multiple high-performance digital TV (DTV) processing chips utilizing Samsung's latest semiconductor technology process nodes. *"Over many years, FlexNoC interconnect IP has helped us accelerate implementation of our digital TV chip designs on our latest semiconductor process nodes. This core interconnect technology is required to develop complex and highly optimized chips in a predictable, low-risk fashion."* SAMSUNG *Jaeyoul Lee, Vice President, **Samsung Electronics*** Samsung first licensed FlexNoC interconnect IP in 2010. Since then, Samsung has used Arteris interconnect IP to enable complex SoC architectures in chips like the Exynos mobile processors and other electronic systems. https://www.arteris.com/press-releases/samsung-lsi-dtv-arteris-ip-flexnoc

U.S. Patent No. 7,769,893 (Goossens)
"Integrated circuit and method for establishing transactions"

'9893 Patent Claim	Samsung Exynos 1280 System on Chip[1]

Arteris Interconnect IP Solution Selected by Samsung for Mobile SoC Deployment

by **Kurt Shuler**, on November 02, 2010

Network-on-Chip (NoC) interconnect technology leader enables higher performance and more cost effective designs for mobile phone systems-on-chip (SoCs)

SUNNYVALE, California — November 2, 2010 — Arteris, Inc., a leading supplier of on-chip interconnect IP solutions, today announced that Samsung Electronics Co., Ltd., has selected Arteris' interconnect solutions for multiple chips within Samsung's mobile SOC products. Samsung chose Arteris interconnect IP to support the high speed inter-chip communication requirements in next generation mobile SOC products.

> *The Arteris interconnect IP offers us a convenient solution to handle the high speed communication needed between our SoC and external modem IC. Our customers will benefit from the lower BOM cost and power consumption as a result of this IP. We look forward to Arteris' interconnect IP helping us shorten development schedules and lower risks associated with compatibility.*

SAMSUNG

Thomas Kim, Vice President, SoC Platform Development, System LSI, ***Samsung Electronics***

https://www.arteris.com/press-releases/pr_2010_nov_02?hsLang=en-us

'9893 Patent Claim	Samsung Exynos 1280 System on Chip[1]
	The Arteris NoC exchanges messages between the plurality of modules via a network in the Exynos SoC. For example, in the Arteris NoC, "[m]ost transactions require the following two-step transfers," including "[a] master send[ing] request packets" and "the slave return[ing] response packets": ***11.3.1.1 Transaction Layer*** The transaction layer is compatible with bus-based transaction protocols used for on-chip communications. It is implemented in NIUs, which are at the boundary of the NoC, and translates between third-party and NTTP protocols. Most transactions require the following two-step transfers: • A master sends request packets. • Then, the slave returns response packets. As shown in Figure 11.1, requests from an initiator are sent through the master NIU's transmit port, Tx, to the NoC request network, where they are routed to the corresponding slave NIU. Slave NIUs, upon reception of request packets

'9893 Patent Claim	Samsung Exynos 1280 System on Chip[1]
	on their receive ports, Rx, translate requests so that they comply with the protocol used by the target third-party IP node. When the target node responds, returning responses are again converted by the slave NIU into appropriate response packets, then delivered through the slave NIU's Tx port to the response network. The network then routes the response packets to the requesting master NIU, which forwards them to the initiator. At the transaction level, NIUs enable multiple protocols to coexist within the same NoC. From the point of view of the NTTP modules, different third-party protocols are just packets moving back and forth across the network.

'9893 Patent Claim	Samsung Exynos 1280 System on Chip[1]

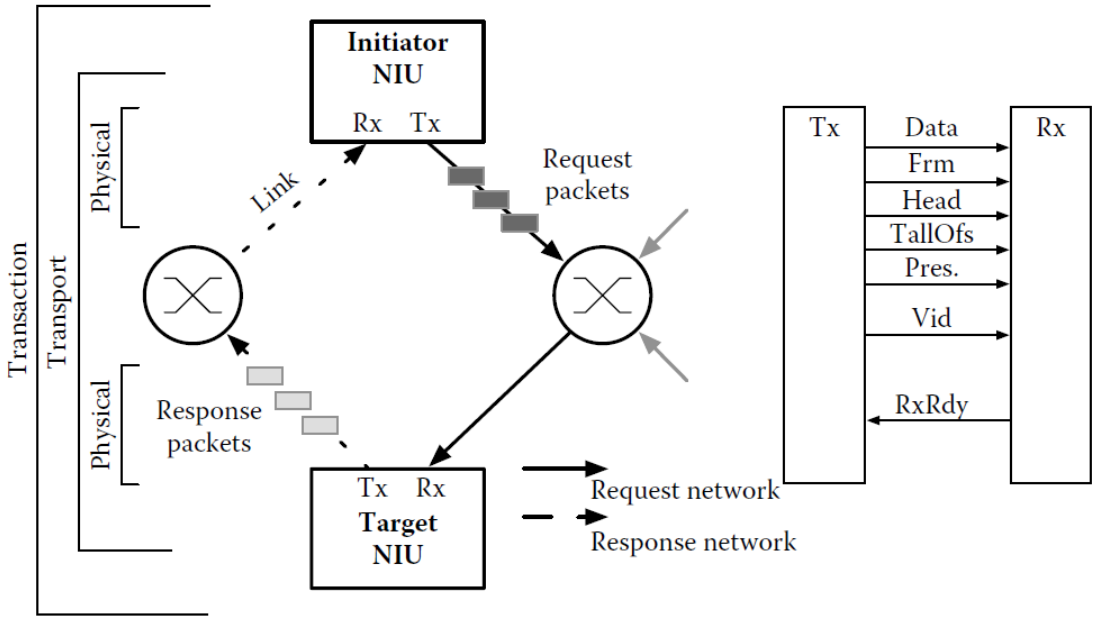
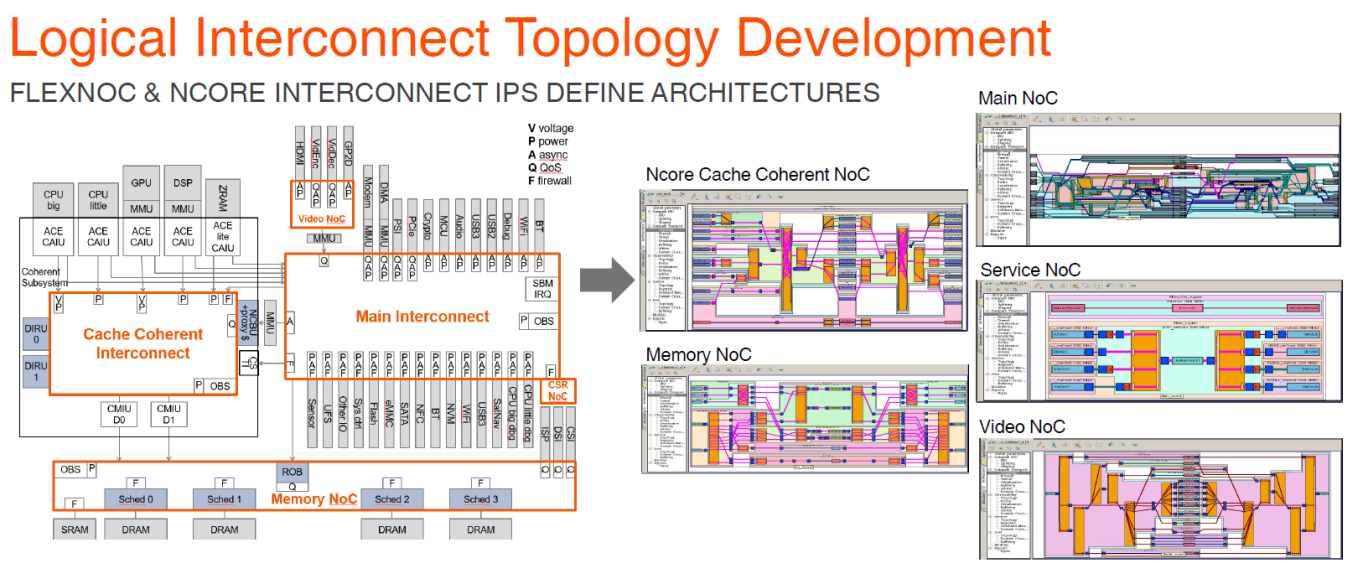


FIGURE 11.1
NTTP protocol layers mapped on NoC units and Media Independent NoC Interface—MINI.

See Networks-On-Chips Theory and Practice, https://vdoc.pub/download/networks-on-chips-theory-and-practice-embedded-multi-core-systems-6f26qivv11f0, at 312-313; *see id* at at 308 (explaining that Chapter 11 of this book describes the function of the Arteris NoC: "In this chapter we will present an MPSoC platform […] using Arteris NoC as communication infrastructure.").

As a further illustration, a large SoC, such as the Exynos SoC may include multiple classes of Arteris NoC network:

U.S. Patent No. 7,769,893 (Goossens)
"Integrated circuit and method for establishing transactions"

'9893 Patent Claim	Samsung Exynos 1280 System on Chip[1]
	 *See* Physical Interconnect Aware Network Optimizer, http://www.ispd.cc/slides/2018/s7_2.pdf, at slide 9.
wherein a message issued by an addressing module M comprises:	Without conceding that the preamble of claim 4 of the '9893 Patent is limiting, a message issued by an addressing module M in the Exynos SoC via the Arteris NoC comprises first information indicative of a location of an addressed message receiving module S within the network and is comprised of (1) a connection identifier identifying two or more message receiving modules S and (2) an identifier of a passive network interface means associated with the addressed message receiving module S, and second information indicative of a particular location within the

'9893 Patent Claim	Samsung Exynos 1280 System on Chip[1]
first information indicative of a location of an addressed message receiving module S within the network and is comprised of (1) a connection identifier identifying two or more message receiving modules S and (2) an identifier of a passive network interface means associated with the addressed message receiving module S, and second information indicative of a particular location within the addressed message receiving module S, such as	addressed message receiving module S, such as a memory, or a register address, either literally or under the doctrine of equivalents. For example, the Arteris NoC used in the Exynos SoC uses Network Interface Units (NIUs), which "translate[] between third-party [OCP, AMBA AHB, APB, and AXI protocols] and NTTP protocols" and in the Arteris NoC, "[m]ost transactions require the following two-step transfers," including "[a] master send[ing] request packets" and "the slave return[ing] response packets": ***11.3.1.1 Transaction Layer*** The transaction layer is compatible with bus-based transaction protocols used for on-chip communications. It is implemented in NIUs, which are at the boundary of the NoC, and translates between third-party and NTTP protocols. Most transactions require the following two-step transfers: • A master sends request packets. • Then, the slave returns response packets. As shown in Figure 11.1, requests from an initiator are sent through the master NIU's transmit port, Tx, to the NoC request network, where they are routed to the corresponding slave NIU. Slave NIUs, upon reception of request packets

'9893 Patent Claim	Samsung Exynos 1280 System on Chip[1]
a memory, or a register address,	on their receive ports, Rx, translate requests so that they comply with the protocol used by the target third-party IP node. When the target node responds, returning responses are again converted by the slave NIU into appropriate response packets, then delivered through the slave NIU's Tx port to the response network. The network then routes the response packets to the requesting master NIU, which forwards them to the initiator. At the transaction level, NIUs enable multiple protocols to coexist within the same NoC. From the point of view of the NTTP modules, different third-party protocols are just packets moving back and forth across the network.

'9893 Patent Claim	Samsung Exynos 1280 System on Chip[1]

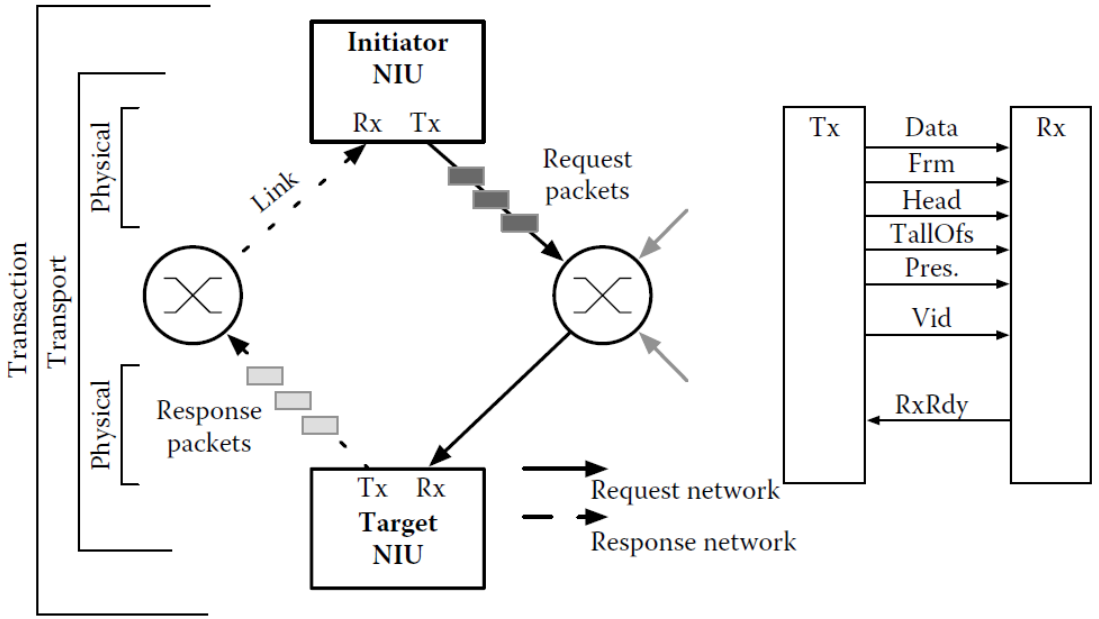


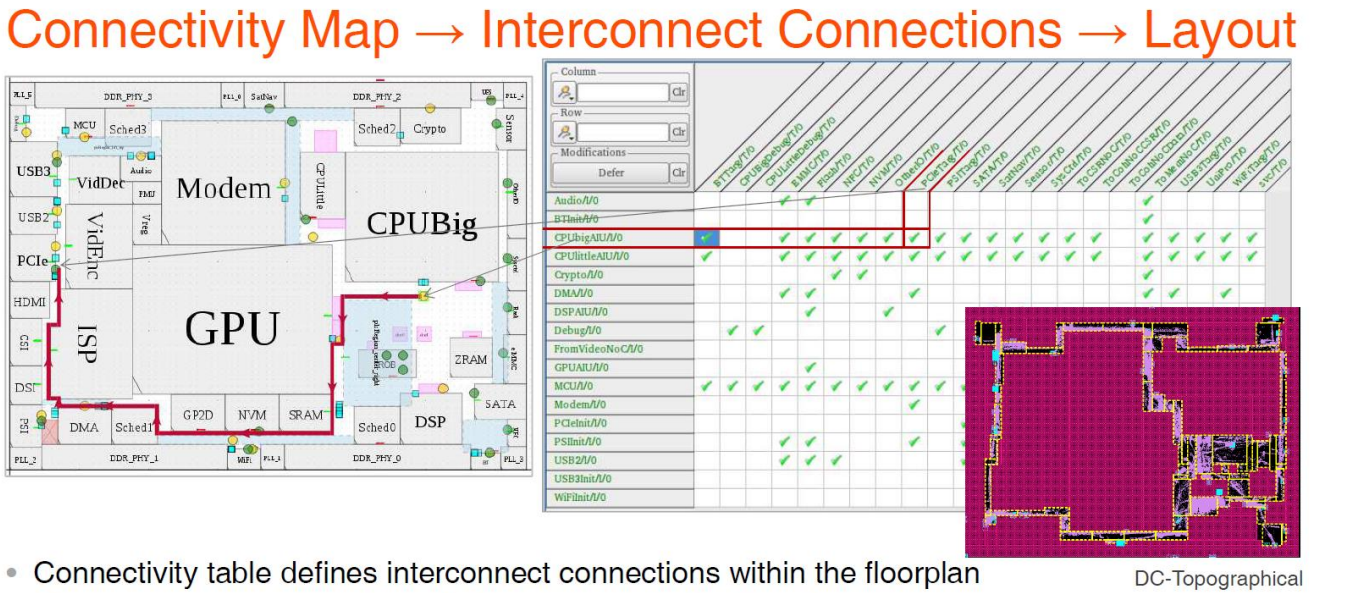
FIGURE 11.1
NTTP protocol layers mapped on NoC units and Media Independent NoC Interface—MINI.

See Networks-On-Chips Theory and Practice, https://vdoc.pub/download/networks-on-chips-theory-and-practice-embedded-multi-core-systems-6f26qivv11f0, at 311, 312-313.

As a further illustration, connections between initiator module NIUs (e.g., "CPUbigAIU/1/0") and two or more target module NIUs (e.g., "ETTarg/T/0," "EMMC/T/0," "Flash/T/0," "NFC/T0," "PCIeTarg/T/0," etc.) within the Arteris NoC may be defined by a connectivity table:

U.S. Patent No. 7,769,893 (Goossens)
"Integrated circuit and method for establishing transactions"

'9893 Patent Claim	Samsung Exynos 1280 System on Chip[1]



See Physical Interconnect Aware Network Optimizer, http://www.ispd.cc/slides/2018/s7_2.pdf, at slide 12.

As a further illustration, the "Arteris NTTP protocol is packet-based" and the packets, which have "header and necker cells [that] contain information relative to routing, payload size, packet type, and the packet target address," are "transported to other parts of the NoC to accomplish the transactions that are required by foreign IP nodes":

'9893 Patent Claim	Samsung Exynos 1280 System on Chip[1]
	11.3.1.2 Transport Layer The Arteris NTTP protocol is packet-based. Packets created by NIUs are transported to other parts of the NoC to accomplish the transactions that are required by foreign IP nodes. All packets are comprised of cells: a header cell, an optional necker cell, and possibly one or more data cells (for packet definition see Figure 11.2; further descriptions of the packet can be found in the next subsection). The header and necker cells contain information relative to routing, payload size, packet type, and the packet target address. Formats for request packets and response packets are slightly different, with the key difference being the presence of an additional cell, the necker, in the request packet to provide detailed addressing information to the target. *See* Networks-On-Chips Theory and Practice, https://vdoc.pub/download/networks-on-chips-theory-and-practice-embedded-multi-core-systems-6f26qivv11f0, at 313. As a further example, the packets sent in the Arteris NoC are "composed of cells that are organized into fields, with each field carrying specific information," including "Slave address" and "Slave offset":

U.S. Patent No. 7,769,893 (Goossens)
"Integrated circuit and method for establishing transactions"

'9893 Patent Claim	Samsung Exynos 1280 System on Chip[1]

Field	Size	Function
Opcode	4 bits/3 bits	Packet type: 4 bits for requests, 3 bits for responses
MstAddr	User Defined	Master address
SlvAddr	User Defined	Slave address
SlvOfs	User Defined	Slave offset
Len	User Defined	Payload length
Tag	User Defined	Tag
Prs	User defined (0 to 2)	Pressure
BE	0 or 4 bits	Byte enables
CE	1 bit	Cell error
Data	32 bits	Packet payload
Info	User Defined	Information about services supported by the NoC
Err	1 bit	Error bit
StartOfs	2 bits	Start offset
StopOfs	2 bits	Stop offset
WrpSize	4 bits	Wrap size
Rsv	Variable	Reserved
CtlId	4 bits/3 bits	Control identifier, for control packets only
CtlInfo	Variable	Control information, for control packets only
EvtId	User defined	Event identifier, for event packets only

U.S. Patent No. 7,769,893 (Goossens)
"Integrated circuit and method for establishing transactions"

'9893 Patent Claim	Samsung Exynos 1280 System on Chip[1]

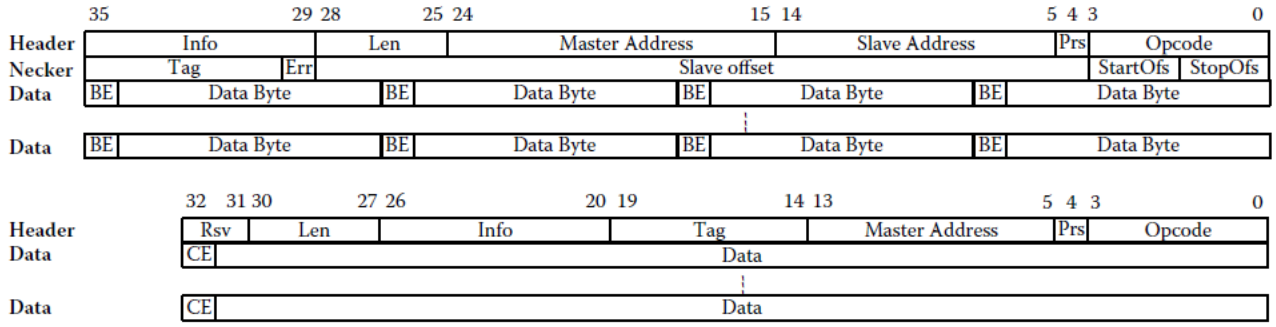


FIGURE 11.2
NTTP packet structure.

Networks-On-Chips Theory and Practice, https://vdoc.pub/download/networks-on-chips-theory-and-practice-embedded-multi-core-systems-6f26qivv11f0, at 313, 314-315.

As a further example, "[i]nitiator NIU units…translate[] AHB transactions AHB transactions into an equivalent NTTP packet sequence, and transports requests and responses to and from a target NIU, that is, slave IP" and the "AHB-to-NTTP unit instantiates a Translation Table for address decoding" with the table "receiv[ing] 32-bit AHB addresses from the NIU and returns the packet header and necker information that is needed to access the NTTP address space: Slave address, Slave offset, Start offset, and the coherency size":

'9893 Patent Claim	Samsung Exynos 1280 System on Chip[1]
	11.3.2.1 Initiator NIU Units Initiator NIU units (the architecture of the AHB initiator is given in Figure 11.4) enable connection between an AMBA-AHB master IP and the NoC. It translates AHB transactions into an equivalent NTTP packet sequence, and transports requests and responses to and from a target NIU, that is, slave IP (slave can be any of the supported protocols). The AHB-to-NTTP unit instantiates a Translation Table for address decoding. This table receives 32-bit AHB addresses from the NIU and returns the packet header and necker information that is needed to access the NTTP address space: Slave address, Slave offset, Start offset, and the coherency size (see Figure 11.2). Whenever the AHB address does not fit the predefined decoding range, the table asserts an error signal that sets the error bit of the corresponding NTTP request packet, for further error handling by the NoC. The translation table is fully user-defined at design time: it must first be completed with its own hardware parameters, then passed to the NIU. Networks-On-Chips Theory and Practice, https://vdoc.pub/download/networks-on-chips-theory-and-practice-embedded-multi-core-systems-6f26qivv11f0, at 317. As further example, "[f]or the AHB target NIU, the AHB address space is mapped from the NTTP address space using the slave offset, the start/stop offset, and the slave address fields, when applicable (from the header of the request packet, Figure 11.2)":

'9893 Patent Claim	Samsung Exynos 1280 System on Chip[1]
	11.3.2.2 Target NIU Units Target NIU units enable connection of a slave IP to the NoC by translating NTTP packet sequences into equivalent packet transactions, and transporting requests and responses to and from targets (the architecture of the AHB Target NIU is given in Figure 11.5). For the AHB target NIU, the AHB address space is mapped from the NTTP address space using the slave offset, the start/stop offset, and the slave address fields, when applicable (from the header of the request packet, Figure 11.2). The AHB address bus is always *Id.* at 318.
the method including the steps of: (a) issuing from said addressing module M a message request including said first information, said second information, and data and/or connection properties to an address	The Arteris NoC utilized by the Exynos SoC issues from said addressing module M a message request including said first information, said second information, and data and/or connection properties to an address translation unit included as part of an active network interface module associated with said addressing module M, either literally or under the doctrine of equivalents. For example, the Arteris NoC used in the Exynos SoC uses Network Interface Units (NIUs), which "translate[] between third-party [OCP, AMBA AHB, APB, and AXI protocols] and NTTP protocols" and in the Arteris NoC, "[m]ost transactions require the following two-step transfers," including "[a] master send[ing] request packets" and "the slave return[ing] response packets":

'9893 Patent Claim	Samsung Exynos 1280 System on Chip[1]
translation unit included as part of an active network interface module associated with said addressing module M,	***11.3.1.1 Transaction Layer*** The transaction layer is compatible with bus-based transaction protocols used for on-chip communications. It is implemented in NIUs, which are at the boundary of the NoC, and translates between third-party and NTTP protocols. Most transactions require the following two-step transfers: • A master sends request packets. • Then, the slave returns response packets. As shown in Figure 11.1, requests from an initiator are sent through the master NIU's transmit port, Tx, to the NoC request network, where they are routed to the corresponding slave NIU. Slave NIUs, upon reception of request packets on their receive ports, Rx, translate requests so that they comply with the protocol used by the target third-party IP node. When the target node responds, returning responses are again converted by the slave NIU into appropriate response packets, then delivered through the slave NIU's Tx port to the response network. The network then routes the response packets to the requesting master NIU, which forwards them to the initiator. At the transaction level, NIUs enable multiple protocols to coexist within the same NoC. From the point of view of the NTTP modules, different third-party protocols are just packets moving back and forth across the network.

'9893 Patent Claim	Samsung Exynos 1280 System on Chip[1]

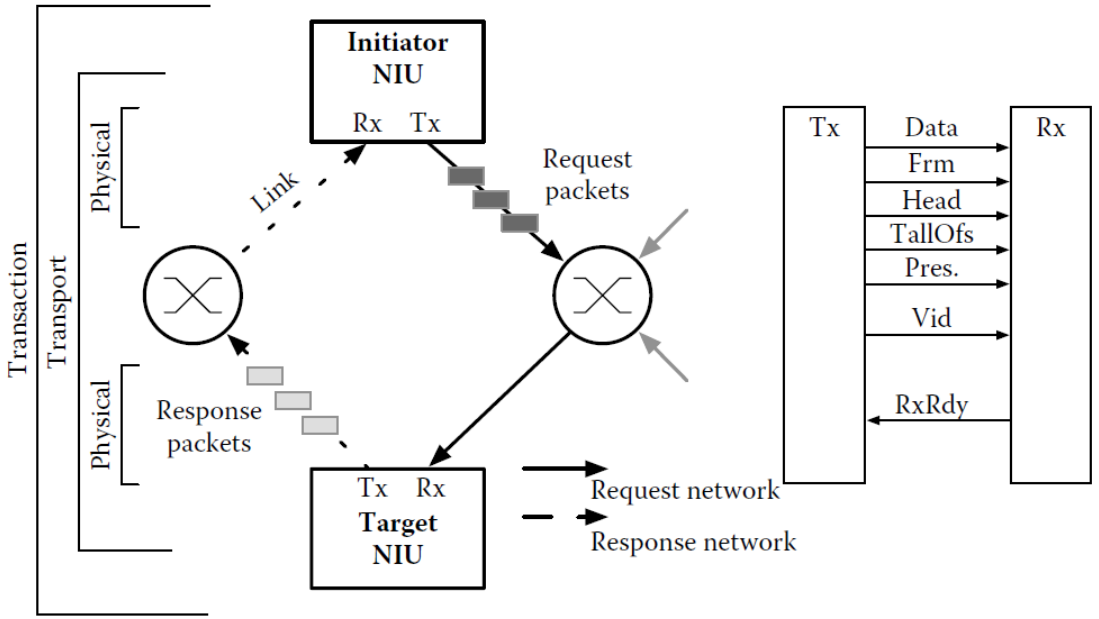
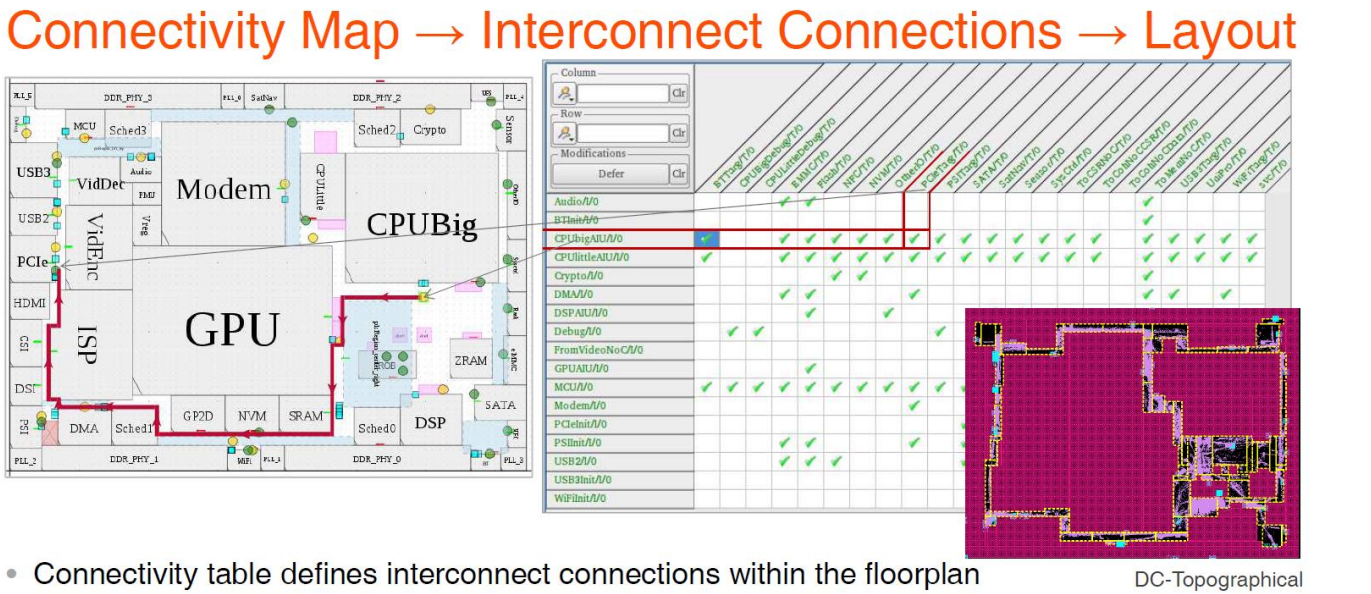


FIGURE 11.1
NTTP protocol layers mapped on NoC units and Media Independent NoC Interface—MINI.

See Networks-On-Chips Theory and Practice, https://vdoc.pub/download/networks-on-chips-theory-and-practice-embedded-multi-core-systems-6f26qivv11f0, at 312-313.

As a further illustration, connections between initiator module NIUs (e.g., "CPUbigAIU/1/0") and two or more target module NIUs (e.g., "ETTarg/T/0," "EMMC/T/0," "Flash/T/0," "NFC/T0," "PCIeTarg/T/0," etc.) within the Arteris NoC may be defined by a connectivity table:

'9893 Patent Claim	Samsung Exynos 1280 System on Chip[1]



See Physical Interconnect Aware Network Optimizer, http://www.ispd.cc/slides/2018/s7_2.pdf, at slide 12.

As a further illustration, the "Arteris NTTP protocol is packet-based" and the packets, which have "header and necker cells [that] contain information relative to routing, payload size, packet type, and the packet target address," are "transported to other parts of the NoC to accomplish the transactions that are required by foreign IP nodes":

'9893 Patent Claim	Samsung Exynos 1280 System on Chip[1]
	11.3.1.2 Transport Layer The Arteris NTTP protocol is packet-based. Packets created by NIUs are transported to other parts of the NoC to accomplish the transactions that are required by foreign IP nodes. All packets are comprised of cells: a header cell, an optional necker cell, and possibly one or more data cells (for packet definition see Figure 11.2; further descriptions of the packet can be found in the next subsection). The header and necker cells contain information relative to routing, payload size, packet type, and the packet target address. Formats for request packets and response packets are slightly different, with the key difference being the presence of an additional cell, the necker, in the request packet to provide detailed addressing information to the target. *Id.* at 313. As yet a further illustration, packets in the Arteris NoC are "delivered as words that are sent along links and "[o]ne link (represented in Figure 11.1) defines the following signals," which include "the current priority of the packet used to define preferred traffic class (or Quality of Service)" and "[f]low control":

maximum cell-width (header, necker, and data cell) and the link-width. One link (represented in Figure 11.1) defines the following signals:

- **Data**—Data word of the width specified at design-time.
- **Frm**—When asserted high, indicates that a packet is being transmitted.
- **Head**—When asserted high, indicates the current word contains a packet header. When the link-width is smaller than single (SGL), the header transmission is split into several word transfers. However, the Head signal is asserted during the first transfer only.
- **TailOfs**—Packet tail: when asserted high, indicates that the current word contains the last packet cell. When the link-width is smaller than single (SGL), the last cell transmission is split into several word transfers. However, the Tail signal is asserted during the first transfer only.
- **Pres.**—Indicates the current priority of the packet used to define preferred traffic class (or Quality of Service). The width is fixed during the design time, allowing multiple pressure levels within the same NoC instance (bits 3–5 in Figure 11.2).
- **Vld**—Data valid: when asserted high, indicates that a word is being transmitted.
- **RxRdy**—Flow control: when asserted high, the receiver is ready to accept word. When de-asserted, the receiver is busy.

This signal set, which constitutes the Media Independent NoC Interface (MINI), is the foundation for NTTP communications.

'9893 Patent Claim	Samsung Exynos 1280 System on Chip[1]

Id. at 313-314.

As a further example, the packets sent in the Arteris NoC are "composed of cells that are organized into fields, with each field carrying specific information," including "Pres," "Slave address" and "Slave offset":

Field	Size	Function
Opcode	4 bits/3 bits	Packet type: 4 bits for requests, 3 bits for responses
MstAddr	User Defined	Master address
SlvAddr	User Defined	Slave address
SlvOfs	User Defined	Slave offset
Len	User Defined	Payload length
Tag	User Defined	Tag
Prs	User defined (0 to 2)	Pressure
BE	0 or 4 bits	Byte enables
CE	1 bit	Cell error
Data	32 bits	Packet payload
Info	User Defined	Information about services supported by the NoC
Err	1 bit	Error bit

'9893 Patent Claim	Samsung Exynos 1280 System on Chip[1]

StartOfs	2 bits	Start offset
StopOfs	2 bits	Stop offset
WrpSize	4 bits	Wrap size
Rsv	Variable	Reserved
CtlId	4 bits/3 bits	Control identifier, for control packets only
CtlInfo	Variable	Control information, for control packets only
EvtId	User defined	Event identifier, for event packets only

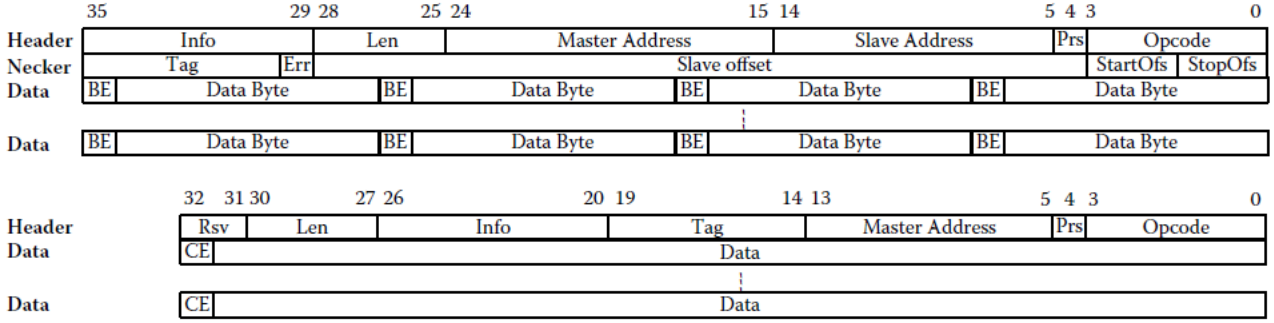


FIGURE 11.2
NTTP packet structure.

Networks-On-Chips Theory and Practice, https://vdoc.pub/download/networks-on-chips-theory-and-practice-embedded-multi-core-systems-6f26qivv11f0, at 313, 314-315.

As a further example, "[i]nitiator NIU units…translate[] AHB transactions AHB transactions into an equivalent NTTP packet sequence, and transports requests and responses to and from a target

'9893 Patent Claim	Samsung Exynos 1280 System on Chip[1]
	NIU, that is, slave IP" and the "AHB-to-NTTP unit instantiates a Translation Table for address decoding" with the table "receiv[ing] 32-bit AHB addresses from the NIU and returns the packet header and necker information that is needed to access the NTTP address space: Slave address, Slave offset, Start offset, and the coherency size": ***11.3.2.1 Initiator NIU Units*** Initiator NIU units (the architecture of the AHB initiator is given in Figure 11.4) enable connection between an AMBA-AHB master IP and the NoC. It translates AHB transactions into an equivalent NTTP packet sequence, and transports requests and responses to and from a target NIU, that is, slave IP (slave can be any of the supported protocols). The AHB-to-NTTP unit instantiates a Translation Table for address decoding. This table receives 32-bit AHB addresses from the NIU and returns the packet header and necker information that is needed to access the NTTP address space: Slave address, Slave offset, Start offset, and the coherency size (see Figure 11.2). Whenever the AHB address does not fit the predefined decoding range, the table asserts an error signal that sets the error bit of the corresponding NTTP request packet, for further error handling by the NoC. The translation table is fully user-defined at design time: it must first be completed with its own hardware parameters, then passed to the NIU. Networks-On-Chips Theory and Practice, https://vdoc.pub/download/networks-on-chips-theory-and-practice-embedded-multi-core-systems-6f26qivv11f0, at 317.

'9893 Patent Claim	Samsung Exynos 1280 System on Chip[1]
	As further example, "[f]or the AHB target NIU, the AHB address space is mapped from the NTTP address space using the slave offset, the start/stop offset, and the slave address fields, when applicable (from the header of the request packet, Figure 11.2)": ***11.3.2.2 Target NIU Units*** Target NIU units enable connection of a slave IP to the NoC by translating NTTP packet sequences into equivalent packet transactions, and transporting requests and responses to and from targets (the architecture of the AHB Target NIU is given in Figure 11.5). For the AHB target NIU, the AHB address space is mapped from the NTTP address space using the slave offset, the start/stop offset, and the slave address fields, when applicable (from the header of the request packet, Figure 11.2). The AHB address bus is always *Id.* at 318. As a further illustration, the Arteris NoC implements Quality of Service (QoS) to "provide[] a regulation mechanism allowing specification of guarantees on some of the parameters related to the traffic"; "QoS, which includes guarantees of throughput and/or latency, is achieved by exploiting the signal pressure embedded into the NTTP packet definition" where the "pressure signal can be generated by the IP itself and is typically linked to a certain level of urgency with which the transaction will have to be completed"; and the "pressure information will be embedded in the NTTP packet at the NIU level":

'9893 Patent Claim	Samsung Exynos 1280 System on Chip[1]
	Quality of Service (QoS). The QoS is a very important feature in the interconnect infrastructures because it provides a regulation mechanism allowing specification of guarantees on some of the parameters related to the traffic. Usually the end users are looking for guarantees on bandwidth and/or end-to-end communication latency. Different mechanisms and strategies have been proposed in the literature. For instance, in Æthereal NoC [11,24] proposed by NXP, a TDMA approach allows the specification of two traffic categories [25]: BE and GT. In the Arteris NoC, the QoS is achieved by exploiting the signal pressure embedded into the NTTP packet definition (Figures 11.1 and 11.2). The pressure

'9893 Patent Claim	Samsung Exynos 1280 System on Chip[1]
	signal can be generated by the IP itself and is typically linked to a certain level of urgency with which the transaction will have to be completed. For example, we can imagine associating the generation of the pressure signal when a certain threshold has been reached in the FIFO of the corresponding IP. This pressure information will be embedded in the NTTP packet at the NIU level: packets that have pressure bits equal to zero will be considered without QoS; packets with a nonzero value of the pressure bit will indicate preferred traffic class.* Such a QoS mechanism offers immediate service to the most urgent inputs and variables, and fair service whenever there are multiple contending inputs of equal urgency (BE). Within switches, arbitration decisions favor preferred packets and allocate remaining bandwidth (after preferred packets are served) fairly to contending packets. When there are contending preferred packets at the same pressure level, arbitration decisions among them are also fair. The Arteris NoC supports the following four different traffic classes:

'9893 Patent Claim	Samsung Exynos 1280 System on Chip[1]
	• **Real time and low latency (RTLL)**—Traffic flows that require the lowest possible latency. Sometimes it is acceptable to have brief intervals of longer latency as long as the average latency is low. Care must be taken to avoid starving other traffic flows as a side effect of pursuing low latency. • **Guaranteed throughput (GT)**—Traffic flows that must maintain their throughput over a relatively long time interval. The actual bandwidth needed can be highly variable even over long intervals. Dynamic pressure is employed for this traffic class. • **Guaranteed bandwidth (GBW)**—Traffic flows that require a guaranteed amount of bandwidth over a relatively long time interval. Over short periods, the network may lag or lead in providing this bandwidth. Bandwidth meters may be inserted onto links in the NoC to regulate these flows, using either of the two methods. If the flow is assigned high pressure, the meter asserts backpressure (flow control) to prevent the flow from exceeding a maximum bandwidth. Alternatively, the meter can modulate the flows pressure (priority) dynamically as needed to maintain an average bandwidth. • **Best effort (BE)**—Traffic flows that do not require guaranteed latency or throughput but have an expectation of fairness.

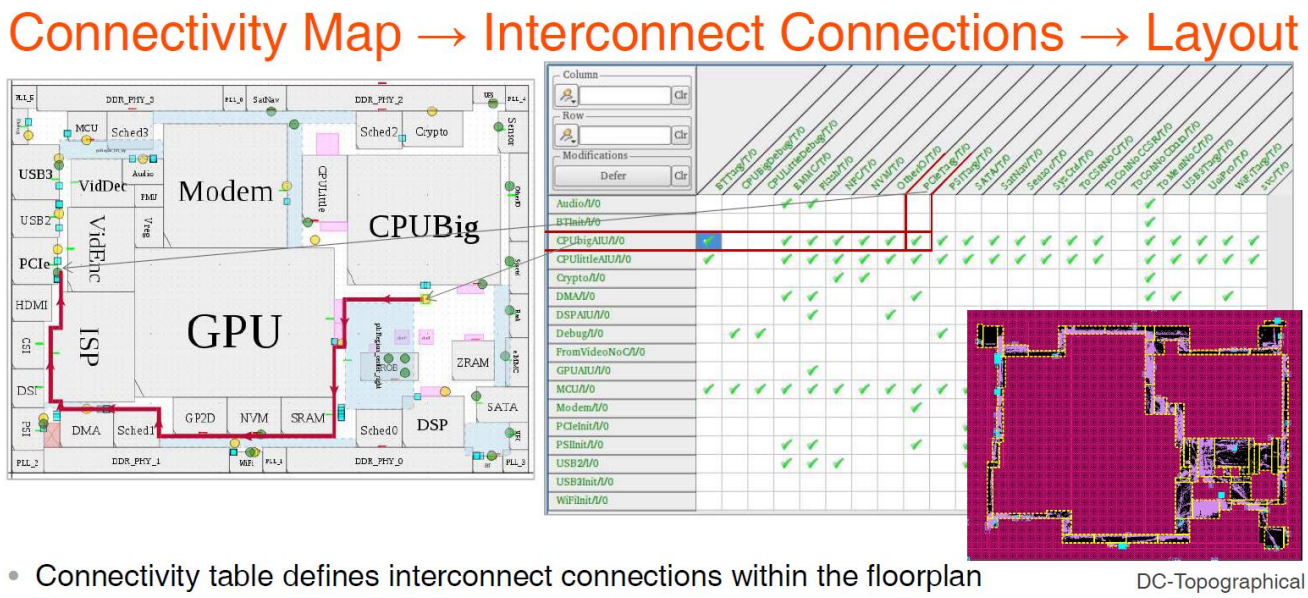
U.S. Patent No. 7,769,893 (Goossens)
"Integrated circuit and method for establishing transactions"

'9893 Patent Claim	Samsung Exynos 1280 System on Chip[1]

* Note that in the NTTP packet, the pressure field allows more then one bit, resulting in multiple levels of preferred traffic.

Networks-On-Chips Theory and Practice, https://vdoc.pub/download/networks-on-chips-theory-and-practice-embedded-multi-core-systems-6f26qivv11f0, at 315-316.

Connections within the Arteris NoC may be defined by a connectivity table:



'9893 Patent Claim	Samsung Exynos 1280 System on Chip[1]

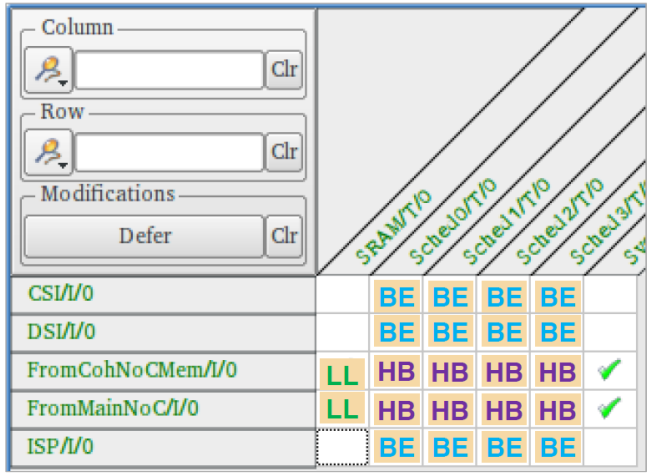
See Physical Interconnect Aware Network Optimizer, http://www.ispd.cc/slides/2018/s7_2.pdf, at slide 12.

As a further illustration, connections within the Arteris NoC may be classified by traffic class and traffic classes, including related to latency, may be mapped onto the Arteris interconnect topology:

Memory NoC:
Interconnect Topology – Traffic Classes

Classify your IP connections per class of traffic:

Best Effort (BE)	Image system
Low Latency (LL)	SRAM
High Bandwidth (HB)	Main/Coherency

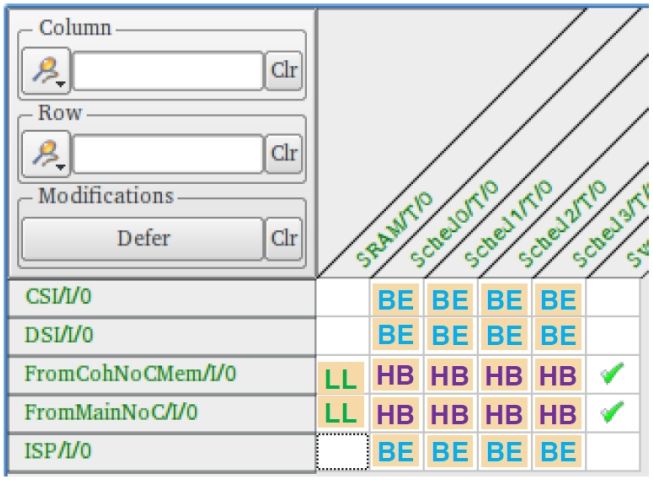


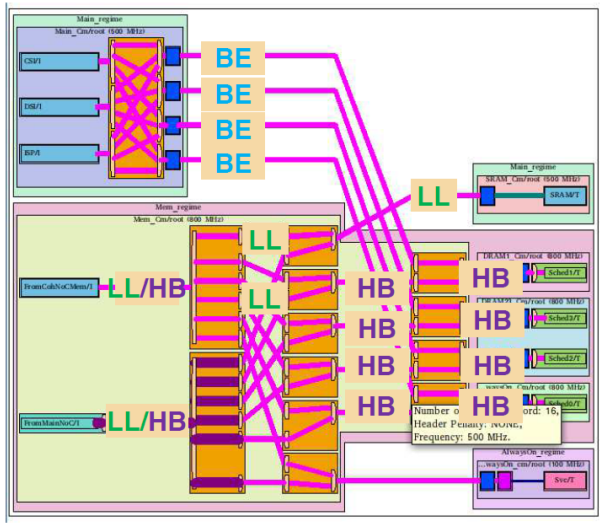
U.S. Patent No. 7,769,893 (Goossens)
"Integrated circuit and method for establishing transactions"

'9893 Patent Claim	Samsung Exynos 1280 System on Chip[1]

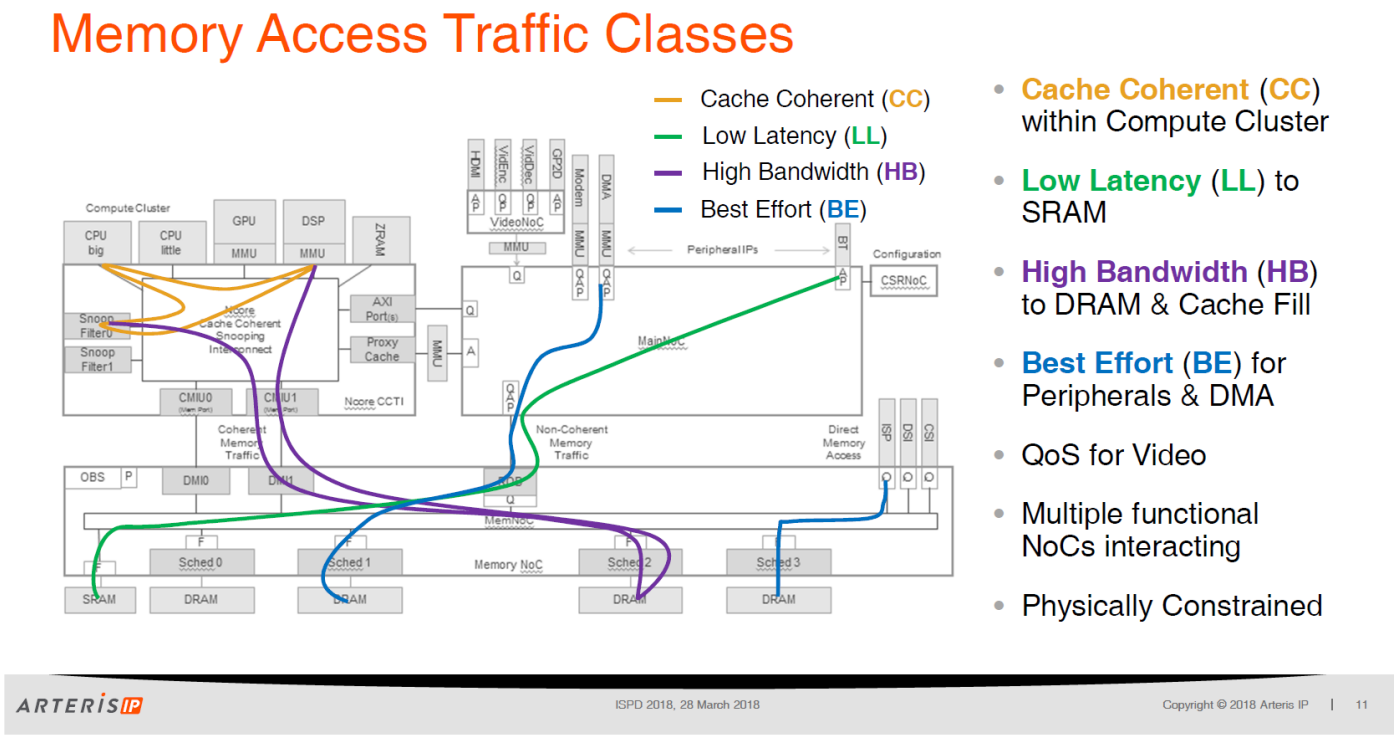
Memory NoC:

Traffic classes are mapped onto logical interconnect topology





ARTERIS IP ISPD 2018, 28 March 2018 Copyright © 2018 Arteris IP | 16

'9893 Patent Claim	Samsung Exynos 1280 System on Chip[1]
	 *See* Physical Interconnect Aware Network Optimizer, http://www.ispd.cc/slides/2018/s7_2.pdf, at slides 11, 13, 16.
(b) arranging, at said address translation unit, the first and the second	The Arteris NoC utilized by the Exynos SoC arranges, at said address translation unit, the first and the second information comprising said issued message as a single address, either literally or under the doctrine of equivalents.

'9893 Patent Claim	Samsung Exynos 1280 System on Chip[1]
information comprising said issued message as a single address,	For example, the Arteris NoC used in the Exynos SoC uses Network Interface Units (NIUs), which "translate[] between third-party [OCP, AMBA AHB, APB, and AXI protocols] and NTTP protocols" and in the Arteris NoC, "[m]ost transactions require the following two-step transfers," including "[a] master send[ing] request packets" and "the slave return[ing] response packets": ***11.3.1.1 Transaction Layer*** The transaction layer is compatible with bus-based transaction protocols used for on-chip communications. It is implemented in NIUs, which are at the boundary of the NoC, and translates between third-party and NTTP protocols. Most transactions require the following two-step transfers: • A master sends request packets. • Then, the slave returns response packets. As shown in Figure 11.1, requests from an initiator are sent through the master NIU's transmit port, Tx, to the NoC request network, where they are routed to the corresponding slave NIU. Slave NIUs, upon reception of request packets

'9893 Patent Claim	Samsung Exynos 1280 System on Chip[1]
	on their receive ports, Rx, translate requests so that they comply with the protocol used by the target third-party IP node. When the target node responds, returning responses are again converted by the slave NIU into appropriate response packets, then delivered through the slave NIU's Tx port to the response network. The network then routes the response packets to the requesting master NIU, which forwards them to the initiator. At the transaction level, NIUs enable multiple protocols to coexist within the same NoC. From the point of view of the NTTP modules, different third-party protocols are just packets moving back and forth across the network.

'9893 Patent Claim	Samsung Exynos 1280 System on Chip[1]

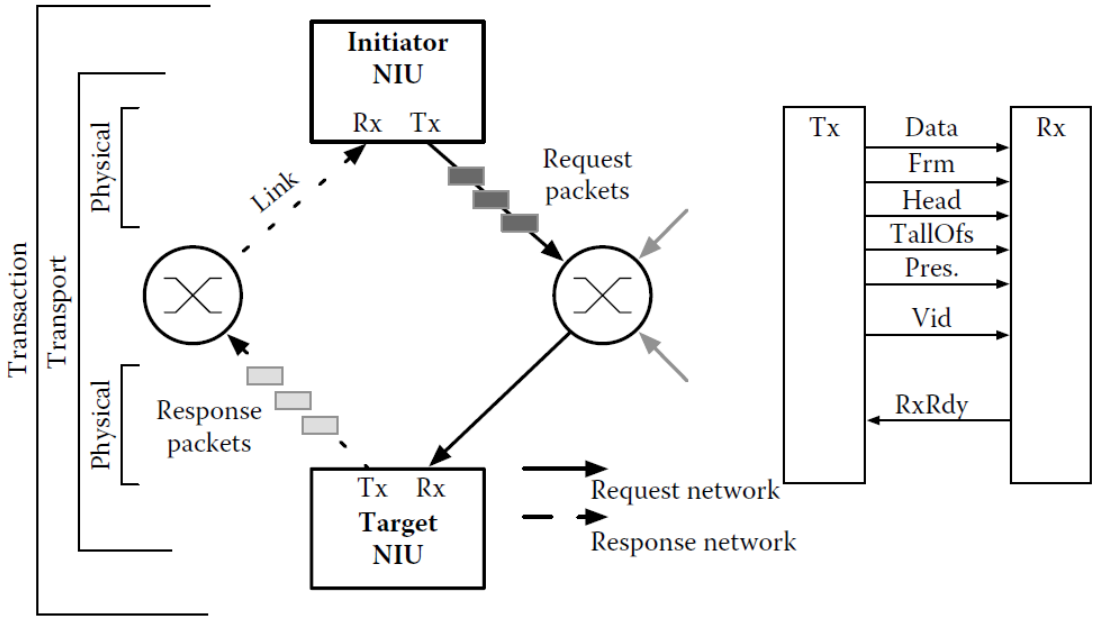


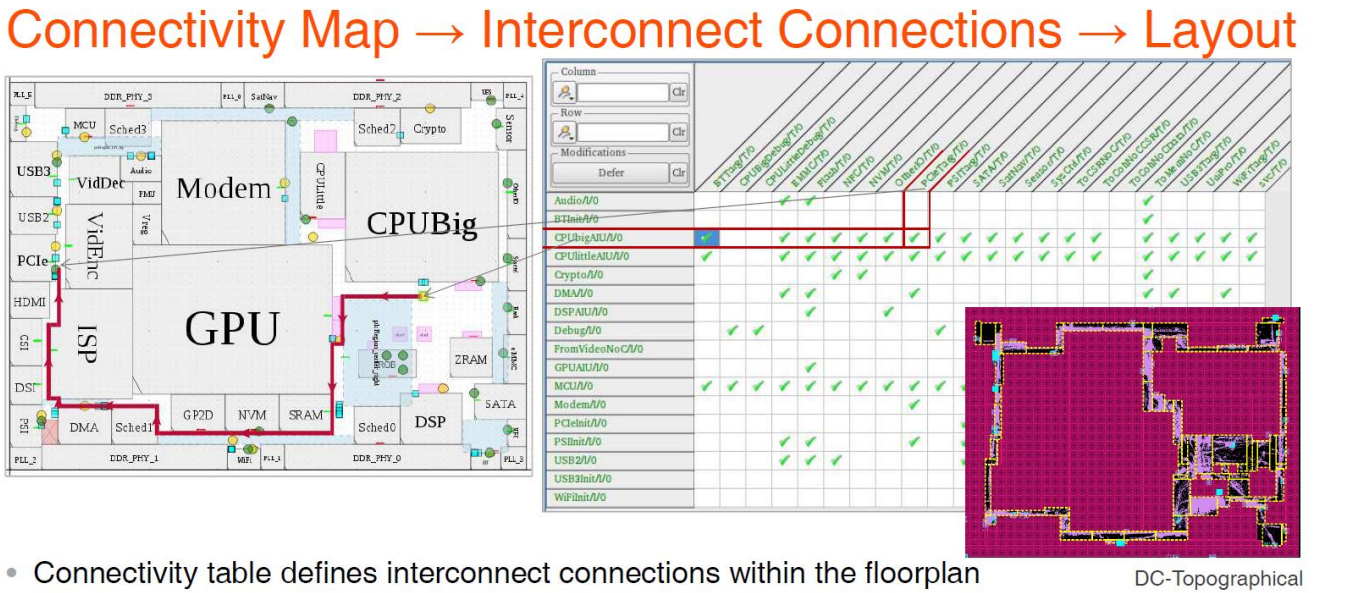
FIGURE 11.1
NTTP protocol layers mapped on NoC units and Media Independent NoC Interface—MINI.

See Networks-On-Chips Theory and Practice, https://vdoc.pub/download/networks-on-chips-theory-and-practice-embedded-multi-core-systems-6f26qivv11f0, at 312-313.

As a further illustration, connections between initiator module NIUs (e.g., "CPUbigAIU/1/0") and two or more target module NIUs (e.g., "ETTarg/T/0," "EMMC/T/0," "Flash/T/0," "NFC/T0," "PCIeTarg/T/0," etc.) within the Arteris NoC may be defined by a connectivity table:

U.S. Patent No. 7,769,893 (Goossens)
"Integrated circuit and method for establishing transactions"

'9893 Patent Claim	Samsung Exynos 1280 System on Chip[1]



See Physical Interconnect Aware Network Optimizer, http://www.ispd.cc/slides/2018/s7_2.pdf, at slide 12.

As a further illustration, the "Arteris NTTP protocol is packet-based" and the packets, which have "header and necker cells [that] contain information relative to routing, payload size, packet type, and the packet target address," are "transported to other parts of the NoC to accomplish the transactions that are required by foreign IP nodes":

'9893 Patent Claim	Samsung Exynos 1280 System on Chip[1]
	11.3.1.2 Transport Layer The Arteris NTTP protocol is packet-based. Packets created by NIUs are transported to other parts of the NoC to accomplish the transactions that are required by foreign IP nodes. All packets are comprised of cells: a header cell, an optional necker cell, and possibly one or more data cells (for packet definition see Figure 11.2; further descriptions of the packet can be found in the next subsection). The header and necker cells contain information relative to routing, payload size, packet type, and the packet target address. Formats for request packets and response packets are slightly different, with the key difference being the presence of an additional cell, the necker, in the request packet to provide detailed addressing information to the target. *Id.* at 313. As a further example, the packets sent in the Arteris NoC are "composed of cells that are organized into fields, with each field carrying specific information," including "Pres," "Slave address" and "Slave offset":

'9893 Patent Claim	Samsung Exynos 1280 System on Chip[1]

Field	Size	Function
Opcode	4 bits/3 bits	Packet type: 4 bits for requests, 3 bits for responses
MstAddr	User Defined	Master address
SlvAddr	User Defined	Slave address
SlvOfs	User Defined	Slave offset
Len	User Defined	Payload length
Tag	User Defined	Tag
Prs	User defined (0 to 2)	Pressure
BE	0 or 4 bits	Byte enables
CE	1 bit	Cell error
Data	32 bits	Packet payload
Info	User Defined	Information about services supported by the NoC
Err	1 bit	Error bit
StartOfs	2 bits	Start offset
StopOfs	2 bits	Stop offset
WrpSize	4 bits	Wrap size
Rsv	Variable	Reserved
CtlId	4 bits/3 bits	Control identifier, for control packets only
CtlInfo	Variable	Control information, for control packets only
EvtId	User defined	Event identifier, for event packets only

U.S. Patent No. 7,769,893 (Goossens)
"Integrated circuit and method for establishing transactions"

'9893 Patent Claim	Samsung Exynos 1280 System on Chip[1]

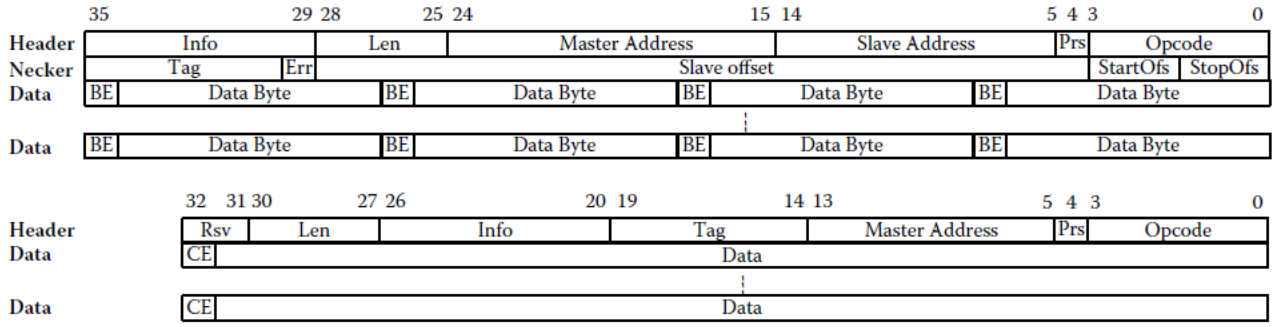


FIGURE 11.2
NTTP packet structure.

Networks-On-Chips Theory and Practice, https://vdoc.pub/download/networks-on-chips-theory-and-practice-embedded-multi-core-systems-6f26qivv11f0, at 313, 314-315.

As a further example, "[i]nitiator NIU units…translate[] AHB transactions AHB transactions into an equivalent NTTP packet sequence, and transports requests and responses to and from a target NIU, that is, slave IP" and the "AHB-to-NTTP unit instantiates a Translation Table for address decoding" with the table "receiv[ing] 32-bit AHB addresses from the NIU and returns the packet header and necker information that is needed to access the NTTP address space: Slave address, Slave offset, Start offset, and the coherency size":

'9893 Patent Claim	Samsung Exynos 1280 System on Chip[1]
	11.3.2.1 Initiator NIU Units Initiator NIU units (the architecture of the AHB initiator is given in Figure 11.4) enable connection between an AMBA-AHB master IP and the NoC. It translates AHB transactions into an equivalent NTTP packet sequence, and transports requests and responses to and from a target NIU, that is, slave IP (slave can be any of the supported protocols). The AHB-to-NTTP unit instantiates a Translation Table for address decoding. This table receives 32-bit AHB addresses from the NIU and returns the packet header and necker information that is needed to access the NTTP address space: Slave address, Slave offset, Start offset, and the coherency size (see Figure 11.2). Whenever the AHB address does not fit the predefined decoding range, the table asserts an error signal that sets the error bit of the corresponding NTTP request packet, for further error handling by the NoC. The translation table is fully user-defined at design time: it must first be completed with its own hardware parameters, then passed to the NIU. Networks-On-Chips Theory and Practice, https://vdoc.pub/download/networks-on-chips-theory-and-practice-embedded-multi-core-systems-6f26qivv11f0, at 317.
(c) determining, at said address translation unit, which message receiving module S is being addressed in said	The Arteris NoC utilized by the Exynos SoC determines, at said address translation unit, which message receiving module S is being addressed in said message request issued from said addressing module M based on said single address, either literally or under the doctrine of equivalents. For example, the Arteris NoC used by the Exynos SoC uses Network Interface Units (NIUs), which "translate[] between third-party [OCP, AMBA AHB, APB, and AXI protocols] and NTTP

'9893 Patent Claim	Samsung Exynos 1280 System on Chip[1]
message request issued from said addressing module M based on said single address, and	protocols" and in the Arteris NoC, "[m]ost transactions require the following two-step transfers," including "[a] master send[ing] request packets" and "the slave return[ing] response packets": ***11.3.1.1 Transaction Layer*** The transaction layer is compatible with bus-based transaction protocols used for on-chip communications. It is implemented in NIUs, which are at the boundary of the NoC, and translates between third-party and NTTP protocols. Most transactions require the following two-step transfers: • A master sends request packets. • Then, the slave returns response packets. As shown in Figure 11.1, requests from an initiator are sent through the master NIU's transmit port, Tx, to the NoC request network, where they are routed to the corresponding slave NIU. Slave NIUs, upon reception of request packets on their receive ports, Rx, translate requests so that they comply with the protocol used by the target third-party IP node. When the target node responds, returning responses are again converted by the slave NIU into appropriate response packets, then delivered through the slave NIU's Tx port to the response network. The network then routes the response packets to the requesting master NIU, which forwards them to the initiator. At the transaction level, NIUs enable multiple protocols to coexist within the same NoC. From the point of view of the NTTP modules, different third-party protocols are just packets moving back and forth across the network.

'9893 Patent Claim	Samsung Exynos 1280 System on Chip[1]

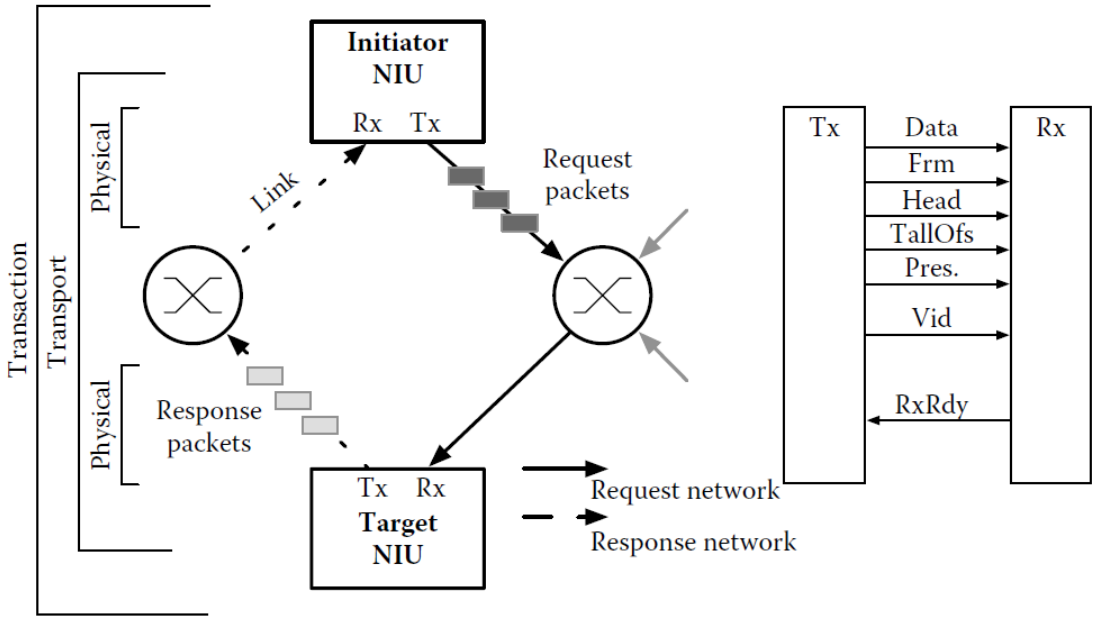


FIGURE 11.1
NTTP protocol layers mapped on NoC units and Media Independent NoC Interface—MINI.

See Networks-On-Chips Theory and Practice, https://vdoc.pub/download/networks-on-chips-theory-and-practice-embedded-multi-core-systems-6f26qivv11f0, at 312-313.

As a further illustration, the "Arteris NTTP protocol is packet-based" and the packets, which have "header and necker cells [that] contain information relative to routing, payload size, packet type, and the packet target address," are "transported to other parts of the NoC to accomplish the transactions that are required by foreign IP nodes":

'9893 Patent Claim	Samsung Exynos 1280 System on Chip[1]
	11.3.1.2 Transport Layer The Arteris NTTP protocol is packet-based. Packets created by NIUs are transported to other parts of the NoC to accomplish the transactions that are required by foreign IP nodes. All packets are comprised of cells: a header cell, an optional necker cell, and possibly one or more data cells (for packet definition see Figure 11.2; further descriptions of the packet can be found in the next subsection). The header and necker cells contain information relative to routing, payload size, packet type, and the packet target address. Formats for request packets and response packets are slightly different, with the key difference being the presence of an additional cell, the necker, in the request packet to provide detailed addressing information to the target. *Id.* at 313. As a further example, the packets sent in the Arteris NoC are "composed of cells that are organized into fields, with each field carrying specific information," including "Slave address" and "Slave offset":

'9893 Patent Claim	Samsung Exynos 1280 System on Chip[1]

Field	Size	Function
Opcode	4 bits/3 bits	Packet type: 4 bits for requests, 3 bits for responses
MstAddr	User Defined	Master address
SlvAddr	User Defined	Slave address
SlvOfs	User Defined	Slave offset
Len	User Defined	Payload length
Tag	User Defined	Tag
Prs	User defined (0 to 2)	Pressure
BE	0 or 4 bits	Byte enables
CE	1 bit	Cell error
Data	32 bits	Packet payload
Info	User Defined	Information about services supported by the NoC
Err	1 bit	Error bit
StartOfs	2 bits	Start offset
StopOfs	2 bits	Stop offset
WrpSize	4 bits	Wrap size
Rsv	Variable	Reserved
CtlId	4 bits/3 bits	Control identifier, for control packets only
CtlInfo	Variable	Control information, for control packets only
EvtId	User defined	Event identifier, for event packets only

U.S. Patent No. 7,769,893 (Goossens)
"Integrated circuit and method for establishing transactions"

'9893 Patent Claim	Samsung Exynos 1280 System on Chip[1]

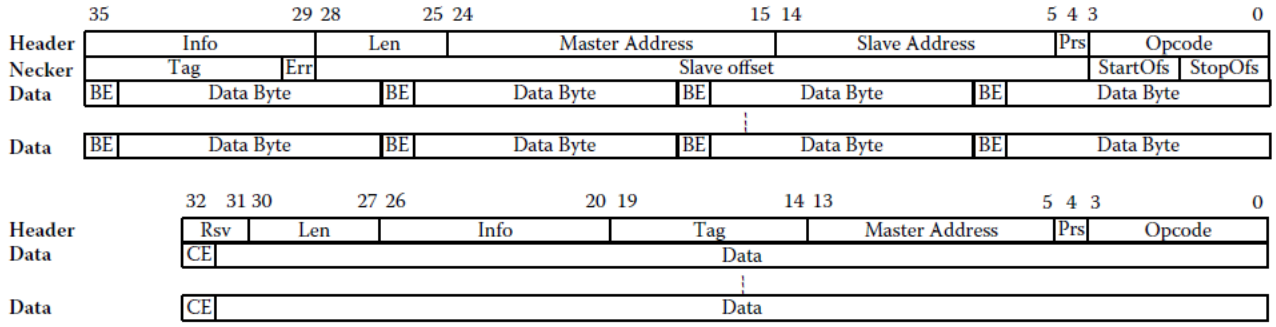


FIGURE 11.2
NTTP packet structure.

Networks-On-Chips Theory and Practice, https://vdoc.pub/download/networks-on-chips-theory-and-practice-embedded-multi-core-systems-6f26qivv11f0, at 313, 314-315.

As a further example, "[i]nitiator NIU units…translate[] AHB transactions AHB transactions into an equivalent NTTP packet sequence, and transports requests and responses to and from a target NIU, that is, slave IP" and the "AHB-to-NTTP unit instantiates a Translation Table for address decoding" with the table "receiv[ing] 32-bit AHB addresses from the NIU and returns the packet header and necker information that is needed to access the NTTP address space: Slave address, Slave offset, Start offset, and the coherency size":

'9893 Patent Claim	Samsung Exynos 1280 System on Chip[1]
	11.3.2.1 Initiator NIU Units Initiator NIU units (the architecture of the AHB initiator is given in Figure 11.4) enable connection between an AMBA-AHB master IP and the NoC. It translates AHB transactions into an equivalent NTTP packet sequence, and transports requests and responses to and from a target NIU, that is, slave IP (slave can be any of the supported protocols). The AHB-to-NTTP unit instantiates a Translation Table for address decoding. This table receives 32-bit AHB addresses from the NIU and returns the packet header and necker information that is needed to access the NTTP address space: Slave address, Slave offset, Start offset, and the coherency size (see Figure 11.2). Whenever the AHB address does not fit the predefined decoding range, the table asserts an error signal that sets the error bit of the corresponding NTTP request packet, for further error handling by the NoC. The translation table is fully user-defined at design time: it must first be completed with its own hardware parameters, then passed to the NIU. Networks-On-Chips Theory and Practice, https://vdoc.pub/download/networks-on-chips-theory-and-practice-embedded-multi-core-systems-6f26qivv11f0, at 317. As further example, "[f]or the AHB target NIU, the AHB address space is mapped from the NTTP address space using the slave offset, the start/stop offset, and the slave address fields, when applicable (from the header of the request packet, Figure 11.2)":

'9893 Patent Claim	Samsung Exynos 1280 System on Chip[1]
	11.3.2.2 Target NIU Units Target NIU units enable connection of a slave IP to the NoC by translating NTTP packet sequences into equivalent packet transactions, and transporting requests and responses to and from targets (the architecture of the AHB Target NIU is given in Figure 11.5). For the AHB target NIU, the AHB address space is mapped from the NTTP address space using the slave offset, the start/stop offset, and the slave address fields, when applicable (from the header of the request packet, Figure 11.2). The AHB address bus is always *Id.* at 318.
(d) further determining, at said address translation unit, the particular location within the addressed message receiving module S based on said single address.	The Arteris NoC utilized by the Exynos SoC further determines, at said address translation unit, the particular location within the addressed message receiving module S based on said single address, either literally or under the doctrine of equivalents. For example, the Arteris NoC uses Network Interface Units (NIUs), which "translate[] between third-party [OCP, AMBA AHB, APB, and AXI protocols] and NTTP protocols" and in the Arteris NoC, "[m]ost transactions require the following two-step transfers," including "[a] master send[ing] request packets" and "the slave return[ing] response packets":

'9893 Patent Claim	Samsung Exynos 1280 System on Chip[1]
	11.3.1.1 Transaction Layer The transaction layer is compatible with bus-based transaction protocols used for on-chip communications. It is implemented in NIUs, which are at the boundary of the NoC, and translates between third-party and NTTP protocols. Most transactions require the following two-step transfers: • A master sends request packets. • Then, the slave returns response packets. As shown in Figure 11.1, requests from an initiator are sent through the master NIU's transmit port, Tx, to the NoC request network, where they are routed to the corresponding slave NIU. Slave NIUs, upon reception of request packets on their receive ports, Rx, translate requests so that they comply with the protocol used by the target third-party IP node. When the target node responds, returning responses are again converted by the slave NIU into appropriate response packets, then delivered through the slave NIU's Tx port to the response network. The network then routes the response packets to the requesting master NIU, which forwards them to the initiator. At the transaction level, NIUs enable multiple protocols to coexist within the same NoC. From the point of view of the NTTP modules, different third-party protocols are just packets moving back and forth across the network.

'9893 Patent Claim	Samsung Exynos 1280 System on Chip[1]

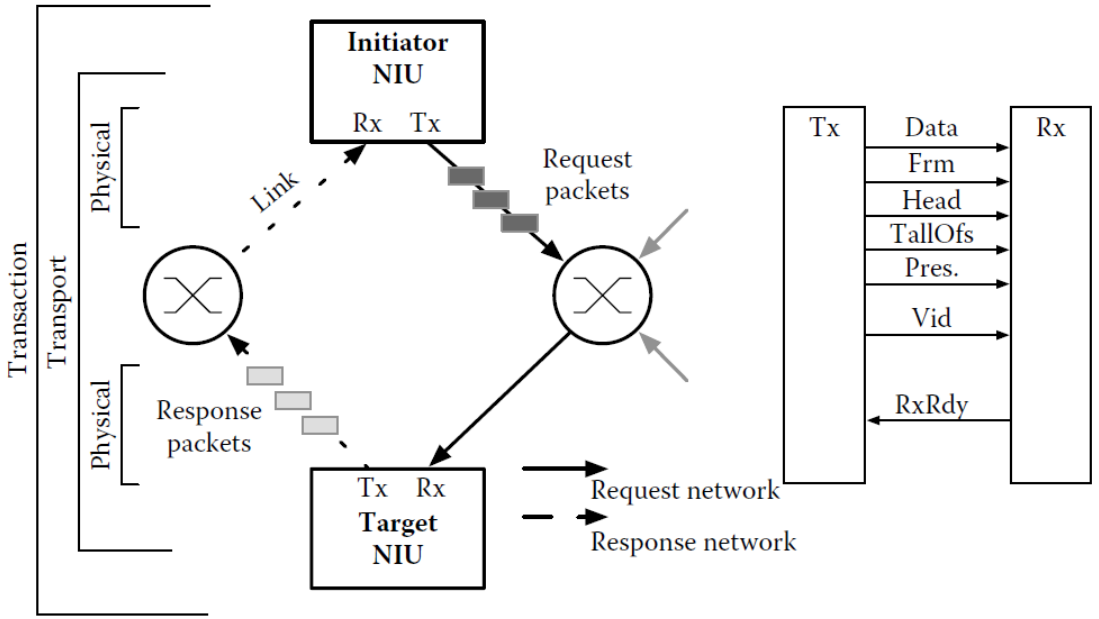


FIGURE 11.1
NTTP protocol layers mapped on NoC units and Media Independent NoC Interface—MINI.

See Networks-On-Chips Theory and Practice, https://vdoc.pub/download/networks-on-chips-theory-and-practice-embedded-multi-core-systems-6f26qivv11f0, at 312-313.

As a further illustration, the "Arteris NTTP protocol is packet-based" and the packets, which have "header and necker cells [that] contain information relative to routing, payload size, packet type, and the packet target address," are "transported to other parts of the NoC to accomplish the transactions that are required by foreign IP nodes":

'9893 Patent Claim	Samsung Exynos 1280 System on Chip[1]
	11.3.1.2 Transport Layer The Arteris NTTP protocol is packet-based. Packets created by NIUs are transported to other parts of the NoC to accomplish the transactions that are required by foreign IP nodes. All packets are comprised of cells: a header cell, an optional necker cell, and possibly one or more data cells (for packet definition see Figure 11.2; further descriptions of the packet can be found in the next subsection). The header and necker cells contain information relative to routing, payload size, packet type, and the packet target address. Formats for request packets and response packets are slightly different, with the key difference being the presence of an additional cell, the necker, in the request packet to provide detailed addressing information to the target. *Id.* at 313. As a further example, the packets sent in the Arteris NoC are "composed of cells that are organized into fields, with each field carrying specific information," including "Slave address" and "Slave offset":

'9893 Patent Claim	Samsung Exynos 1280 System on Chip[1]

Field	Size	Function
Opcode	4 bits/3 bits	Packet type: 4 bits for requests, 3 bits for responses
MstAddr	User Defined	Master address
SlvAddr	User Defined	Slave address
SlvOfs	User Defined	Slave offset
Len	User Defined	Payload length
Tag	User Defined	Tag
Prs	User defined (0 to 2)	Pressure
BE	0 or 4 bits	Byte enables
CE	1 bit	Cell error
Data	32 bits	Packet payload
Info	User Defined	Information about services supported by the NoC
Err	1 bit	Error bit
StartOfs	2 bits	Start offset
StopOfs	2 bits	Stop offset
WrpSize	4 bits	Wrap size
Rsv	Variable	Reserved
CtlId	4 bits/3 bits	Control identifier, for control packets only
CtlInfo	Variable	Control information, for control packets only
EvtId	User defined	Event identifier, for event packets only

'9893 Patent Claim	Samsung Exynos 1280 System on Chip[1]

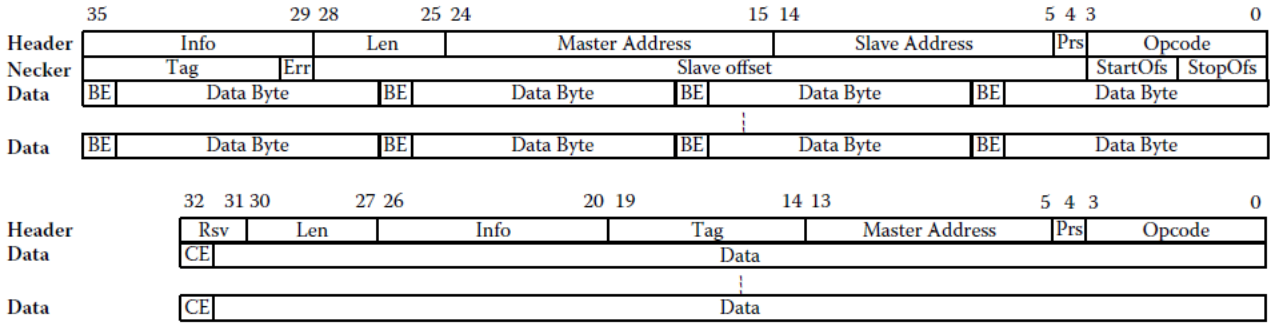


FIGURE 11.2
NTTP packet structure.

Networks-On-Chips Theory and Practice, https://vdoc.pub/download/networks-on-chips-theory-and-practice-embedded-multi-core-systems-6f26qivv11f0, at 313, 314-315.

As a further example, "[i]nitiator NIU units…translate[] AHB transactions AHB transactions into an equivalent NTTP packet sequence, and transports requests and responses to and from a target NIU, that is, slave IP" and the "AHB-to-NTTP unit instantiates a Translation Table for address decoding" with the table "receiv[ing] 32-bit AHB addresses from the NIU and returns the packet header and necker information that is needed to access the NTTP address space: Slave address, Slave offset, Start offset, and the coherency size":

'9893 Patent Claim	Samsung Exynos 1280 System on Chip[1]
	11.3.2.1 Initiator NIU Units Initiator NIU units (the architecture of the AHB initiator is given in Figure 11.4) enable connection between an AMBA-AHB master IP and the NoC. It translates AHB transactions into an equivalent NTTP packet sequence, and transports requests and responses to and from a target NIU, that is, slave IP (slave can be any of the supported protocols). The AHB-to-NTTP unit instantiates a Translation Table for address decoding. This table receives 32-bit AHB addresses from the NIU and returns the packet header and necker information that is needed to access the NTTP address space: Slave address, Slave offset, Start offset, and the coherency size (see Figure 11.2). Whenever the AHB address does not fit the predefined decoding range, the table asserts an error signal that sets the error bit of the corresponding NTTP request packet, for further error handling by the NoC. The translation table is fully user-defined at design time: it must first be completed with its own hardware parameters, then passed to the NIU. Networks-On-Chips Theory and Practice, https://vdoc.pub/download/networks-on-chips-theory-and-practice-embedded-multi-core-systems-6f26qivv11f0, at 317. As further example, "[f]or the AHB target NIU, the AHB address space is mapped from the NTTP address space using the slave offset, the start/stop offset, and the slave address fields, when applicable (from the header of the request packet, Figure 11.2)":

'9893 Patent Claim	Samsung Exynos 1280 System on Chip[1]
	11.3.2.2 Target NIU Units Target NIU units enable connection of a slave IP to the NoC by translating NTTP packet sequences into equivalent packet transactions, and transporting requests and responses to and from targets (the architecture of the AHB Target NIU is given in Figure 11.5). For the AHB target NIU, the AHB address space is mapped from the NTTP address space using the slave offset, the start/stop offset, and the slave address fields, when applicable (from the header of the request packet, Figure 11.2). The AHB address bus is always *Id.* at 318.